Argonne NATIONAL LABORATORY

... for a brighter future

Advances in ELEGANT

Michael Borland, Yusong Wang, Hairong Shang

Accelerator Systems Division
Argonne National Laboratory

December 2, 2008

U.S. Department of Energy

THE UNIVERSITY OF CHICAGO

Office of Science
U.S. DEPARTMENT OF ENERGY

A U.S. Department of Energy laboratory managed by The University of Chicago

Introduction

- Essential goal of this work is to support LCLS commissioning, operation, and optimization with fast, high-fidelity modeling tools
- Summary of ANL's tasks
  - Finish parallelization of relevant parts of **elegant**, a trusted code for such modeling
  - Develop robust interfaces among suite of codes involved in FEL modeling
    - *IMPACT (gun and linac modeling)*
    - *elegant (accelerator modeling and optimization)*
    - *GENESIS and GINGER (FEL modeling)*
  - Develop integrated graphical user interface to provide on-demand, high-fidelity modeling of data and experiments
    - *Selection of codes, algorithms, detail level*
    - *Utilizes data drawn from the control system*
    - *Utilizes high-performance computing resources*
  - Develop optimizer based on genetic algorithm to provide guidance on FEL performance improvement.

Simulation of Microbunching Instability

- The microbunching instability in FEL driver linacs is an important design and operation issue
  - Instability is driven by longitudinal space charge, coherent synchrotron radiation, and non-zero R56 of bunch compressors
- Primary goal of simulations is to determine the microbunching gain curve
  - How much is an initial small density modulation amplified in passing through the system, as a function of wavelength of the modulation?
- We used the FERMI@ELETTRA lattice for testing purposes due to its availability and similarity to LCLS lattice
  - Up-to-date LCLS lattice wasn't available in timely fashion

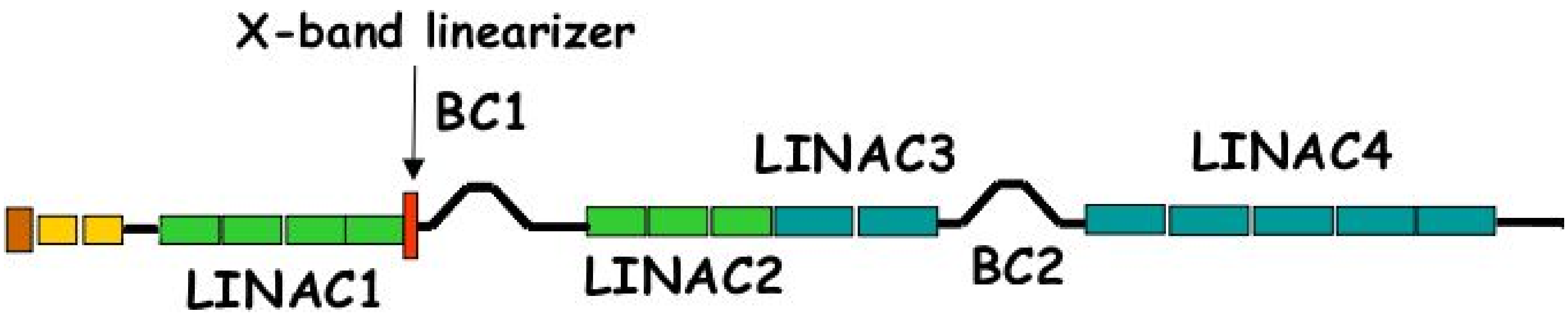
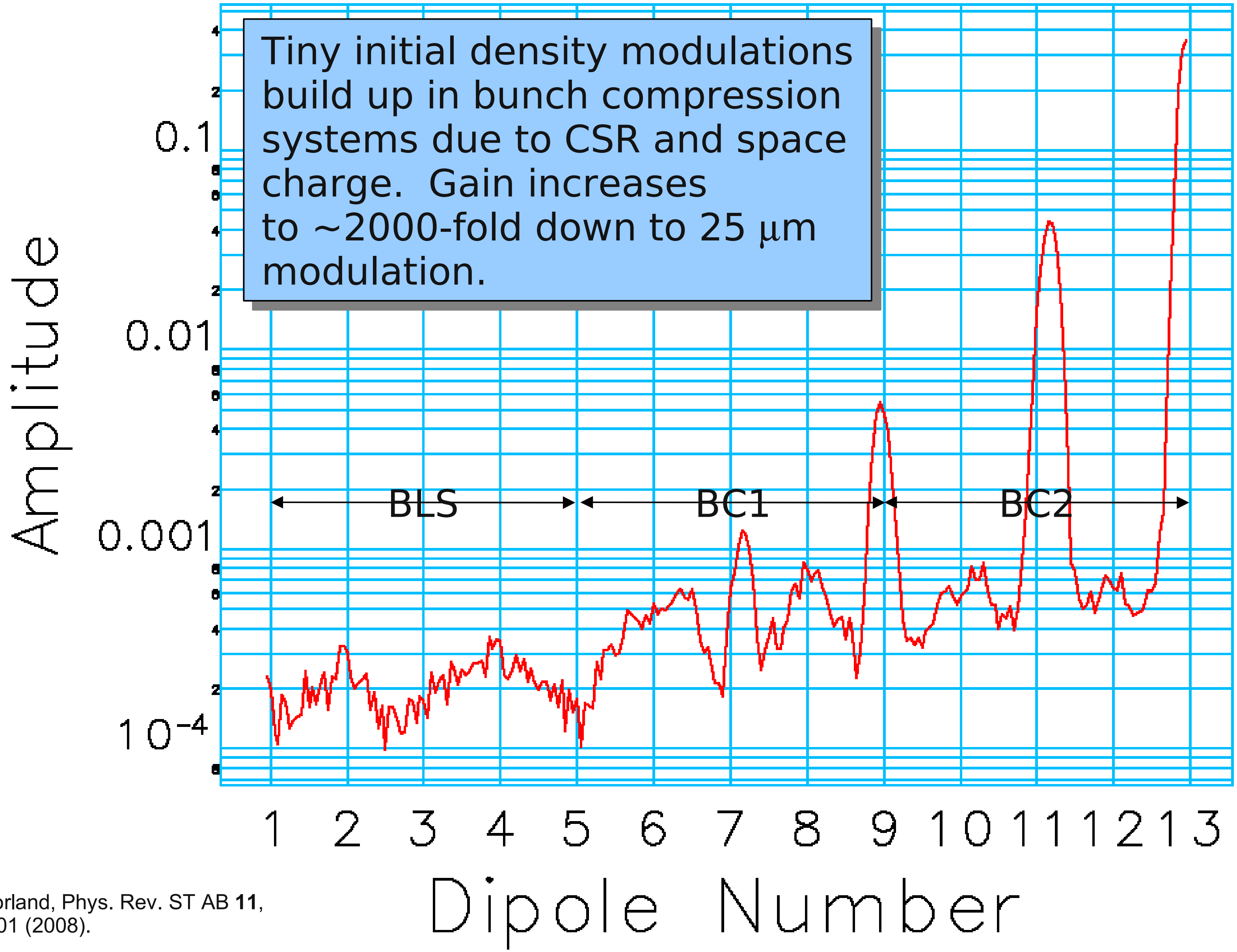


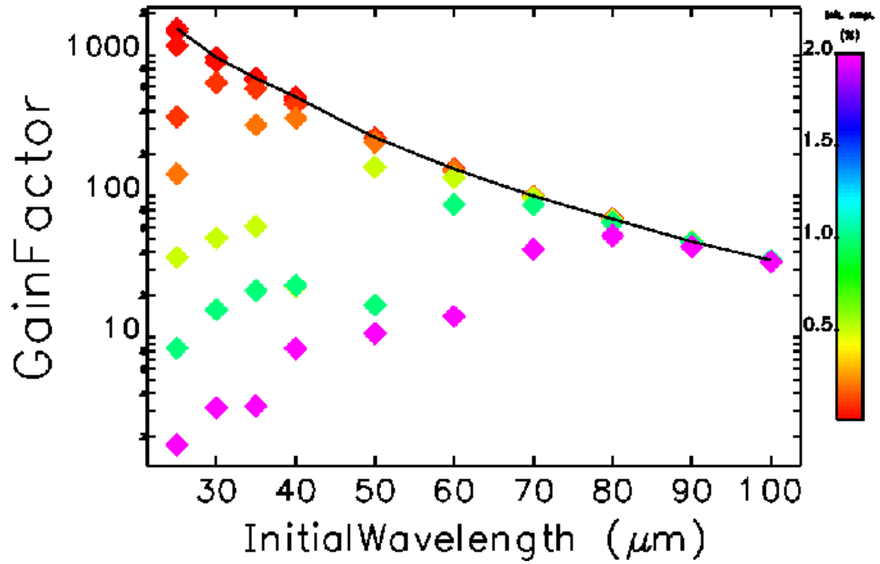
Figure from S. DiMitri *et al*., Proc. 2008 EPAC.

FERMI Microbunching Instability Simulated with elegant



M. Borland, Phys. Rev. ST AB **11**, 030701 (2008).

Reliable Gain Curve Computation Requires Careful Modeling



- Considerations[1,2]:
  - Optimization of initial modulation depth to avoid non-linearity
  - Variation of number of particles to ensure convergence
  - Low-pass filtering of particle distribution in computations to control noise growth
  - Detection of modulation signal in output histograms using NAFF
- For 25 micron modulation in FERMI, need ~20MP, 3000 bins, 0.01% modulation
- Released version of Pelegant can handle up to 60 MP
  - Improved memory management

[1]Z.. Huang *et al.*, Phys. Rev. ST AB **7**, 074401 (2004).
[2]M. Borland, Phys. Rev. ST AB 11, 030701 (2008).

Probing Shorter Wavelengths Important

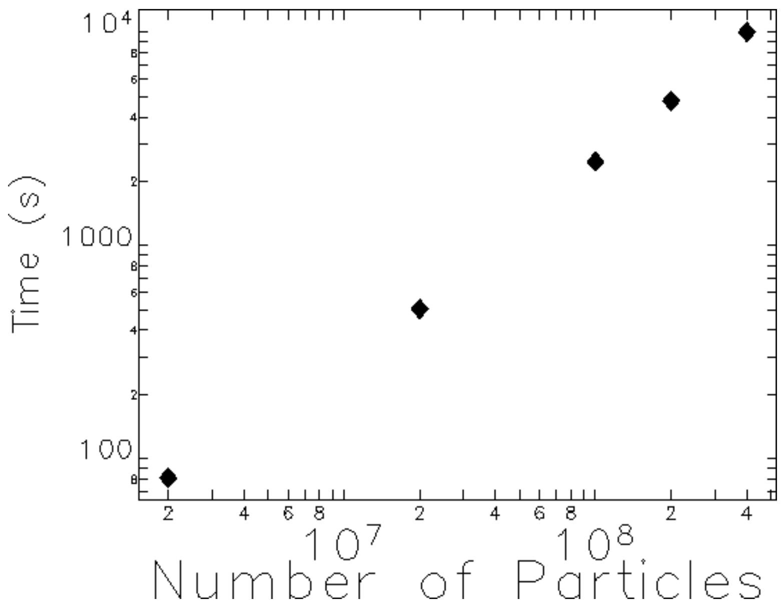
- In LCLS commissioning[1], evidence has been found of the microbunching instability, apparently at optical wavelengths
  - Has serious impact on LCLS diagnostics
  - Need to push modeling into sub-micron wavelengths
- LCLS and other facilities propose[2] a laser/undulator beam heater to impose an energy modulation on the beam
  - Predicted to suppress the microbunching instability
  - Modulates beam energy at ~1 micron wavelength
- 1 micron wavelength requires about 500 million particles
  - Would like to push to 1.5 billion
- To do this requires two things
  - Changing the architecture of parallel **elegant**
  - Parallelizing SDDS-based I/O used by **elegant**

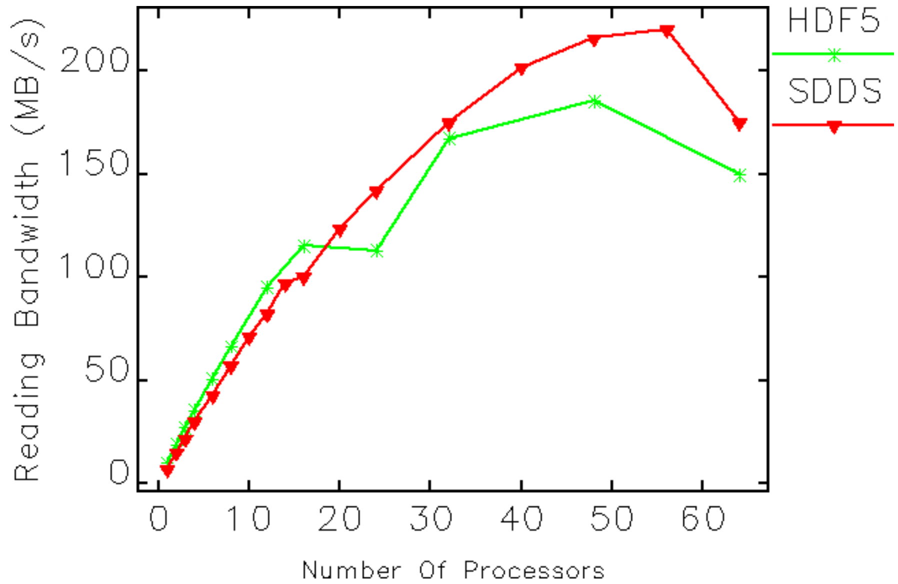
[1]K. Bane *et al*., Proc. PAC07, 807-809 (2007).
[2]Z. Huang *et al*., Phys. Rev. ST AB **7**, 074401 (2004).

Architecture of Parallel elegant

- The original parallelization was an expedient approach
  - Particle-based decomposition only
  - Master performs all I/O
  - Master performs particle generation
  - Master may gather/scatter to perform serial operations, e.g.,
    - *Output*
    - *Elements we didn't get around to parallelizing*
- Benefits:
  - Very useful parallel version in about six months
  - Gradual parallelization of the code without impeding on-going development
- Problem:
  - Master was a memory and I/O bottleneck
  - Limited to about 60M particles (16GB RAM)
- As a result we have reworked the parallelization to eliminate the central role for the master processor
  - Slaves perform I/O and particle generation
  - Not limited to particle-based decomposition for tracking
- This required parallelizing the SDDS-based I/O used by **elegant**

Early Test Results with New Version





- With new version[1], demonstrated 400 MP on 100 nodes
  - Parallelized SDDS I/O
  - Eliminated role of master node in particle management
- Early tests (left) show SDDS I/O performance is comparable to HDF5
  - Test was designed to favor HDF5
  - Performed on Jazz using PVFS

[1]Y. Wang, H. Shang, M. Borland

Why Use SDDS I/O ?

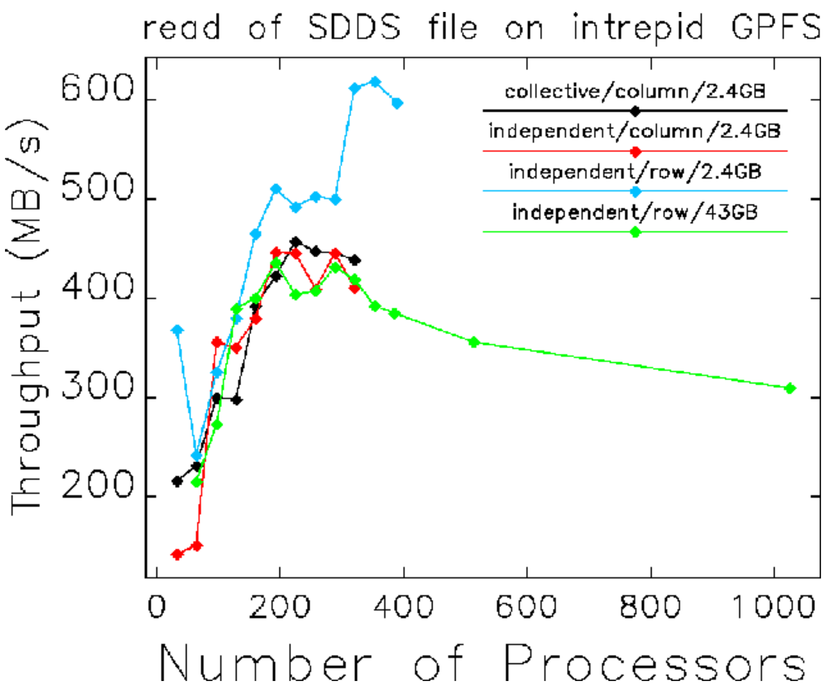
- SDDS (Self Describing Data Sets) refers to
  - A self-describing file protocol developed at ANL starting in 1994
  - A set of general-purpose programs that work with SDDS files
    - *“SDDS Toolkit” for data analysis, manipulation, and display*
    - *“SDDS/EPICS Toolkit” for control-system applications*
- Advantages
  - Provides generic pre- and post-processing tools for simulation codes
  - Supports user scripting
  - Makes writing/upgrading simulations easier
    - *No need to create/modify custom pre- or post-processors*
  - Improves robustness
    - *Programs can detect presence, data type, units of data*
  - Makes it easier to work with multiple codes
    - *Start-to-end simulation*
- Features
  - Highly portable: Linux/UNIX, Windows, OS-X
  - ASCII or binary data storage option
  - Users create custom data analysis methods using pipelines of SDDS tools
  - Libraries support C/C++, FORTRAN, Java, MATLAB, Tcl, Python

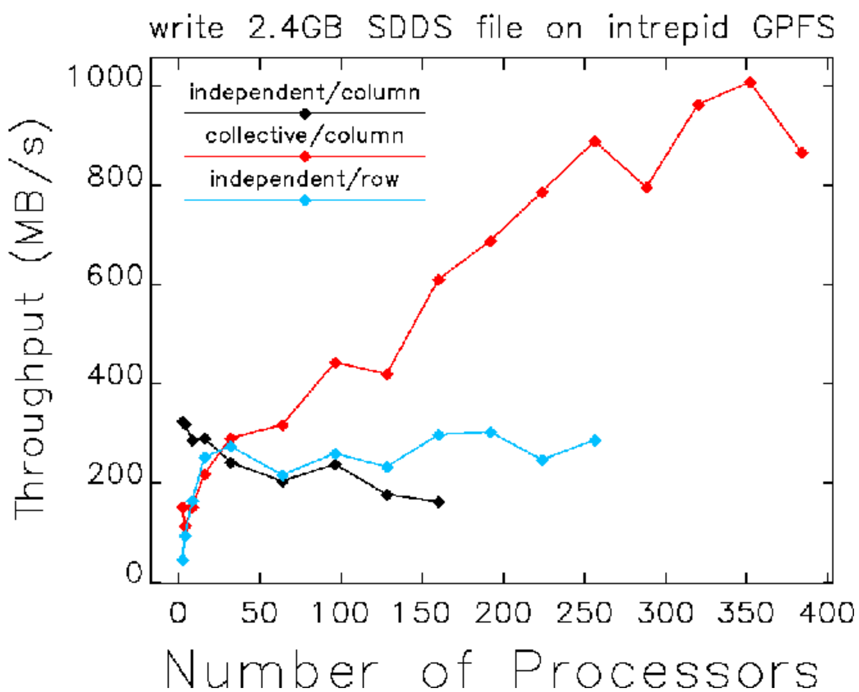
What Codes Use SDDS I/O ?

- **elegant** is the most widely-used SDDS-compliant code
  – SDDS is deeply engrained in how **elegant** is written and used
- **shower** is an interface to EGS4 for electron/gamma shower simulation
- **spiffe** is a 2.5 D PIC code for rf gun simulation
- We have SDDS-based scripts that convert to/from ASTRA and IMPACT-T particle distributions and **elegant** distributions
- **sddsanalyzebeam** analyzes phase space output from **elegant**
- **sddsmatchtwiss** transforms phase space coordinates from **elegant**
- **sddsbrightness** and **sddsurgent** compute properties of synchrotron radiation based on distributions from **elegant**
- **clinchor** computes single- and coupled-bunch growth rates due to HOMs
- **haissinski** computes potential well distortion
- **csrImpedance** computes shielded CSR impedance for use by **elegant**
- **touschekLifetime** computes Touschek lifetime using data from **elegant**
- URMEL/APS is an SDDS-compliant version of the URMEL code for cavity mode computation (output accepted by **elegant** or **clinchor**)
- ABCI/APS provides SDDS wake data that **elegant** accepts
- MAFIA/APS provides SDDS data that (after post-processing) **elegant** accepts
- We have an SDDS-compliant version of the FEL code GENESIS
- Our goal is to never have to manually translate data between two codes

Recent Parallel SDDS I/O Tests on BlueGene

- ANL recently installed a 100 terraflop BlueGene/P
  - Developer workshop held to give opportunity to port and test codes with up to 1024 processors
  - We took the opportunity to test and optimize SDDS I/O
- BlueGene/P uses IBM's General Parallel File System (GPFS)
- Tests are incomplete but performance is good
  - With ~100 or more processors, getting >200 MB/s throughput
  - Implies reading/writing 1B particle file in <5 minutes





Plans

- Further testing and debugging needed for parallel SDDS
- **elegant** is in constant development, so latest parallel version diverged from official version
  - Presently using other funds to merge with official version
  - Expect completion within a few months
- Parallel SDDS I/O in column mode requires other applications to be upgraded to SDDS3
  - This work is in progress (other funding)
- SDDS Toolkit can benefit from multiprocessor machines
  - Will parallelize selected applications using OpenMP (other funding)
- Perform production runs for LCLS with up to 500 MP to get gain curve down to ~1 micron
- Develop IMPACT-T/**elegant**-based GUI application for LCLS modeling
  - Ability to model experiments (e.g., scans) conveniently
  - Intelligent segmenting of runs to improve performance
  - Ability to display expected results at diagnostics
  - Eventually take settings directly from control system
- Employ genetic optimizer to perform start-to-end optimization of LCLS, including FEL modeling